**Урок 9: Обработка исключений**

**Повторение прошлого материала (10 минут):**

“Что мы изучали на прошлом уроке? Кто может напомнить, как работать с массивами и коллекциями? Какие преимущества у ArrayList по сравнению с массивами?

**Цели и задачи урока:**

“Сегодня мы разберем что такое исключения и как их обрабатывать в Java”

**Познание нового (25 минут):**

1. Что такое исключения:

“Исключение — это событие, которое возникает во время выполнения программы и нарушает ее нормальное выполнение. В Java исключения представляют собой объекты, описывающие ошибку или непредвиденное поведение программы.”

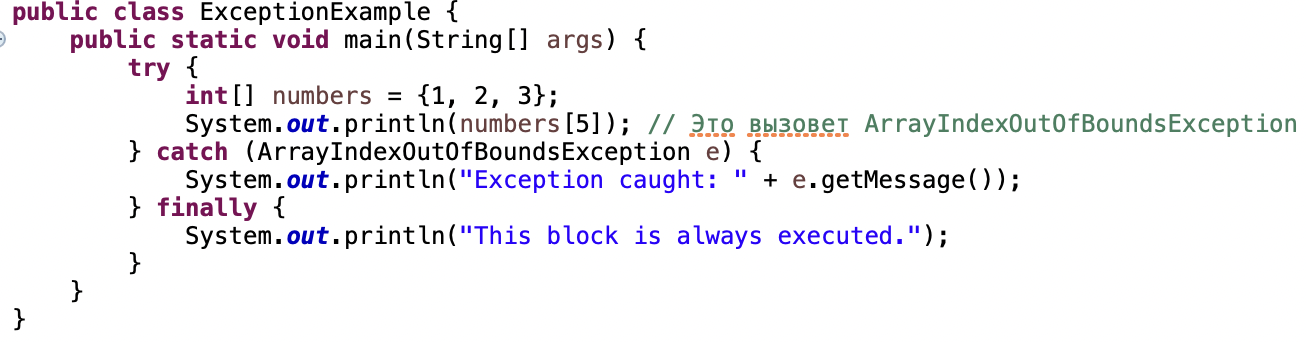
2. Типы исключений:

“В Java существует два основных типа исключений: проверяемые (checked) и непроверяемые (unchecked) исключения. Проверяемые исключения — это те, которые должны быть обработаны явно в коде. Непроверяемые исключения возникают в результате ошибок времени выполнения.”

3.Ключевые слова try, catch, finally:

“Блоки try, catch и finally используются для обработки исключений.”

Пример:

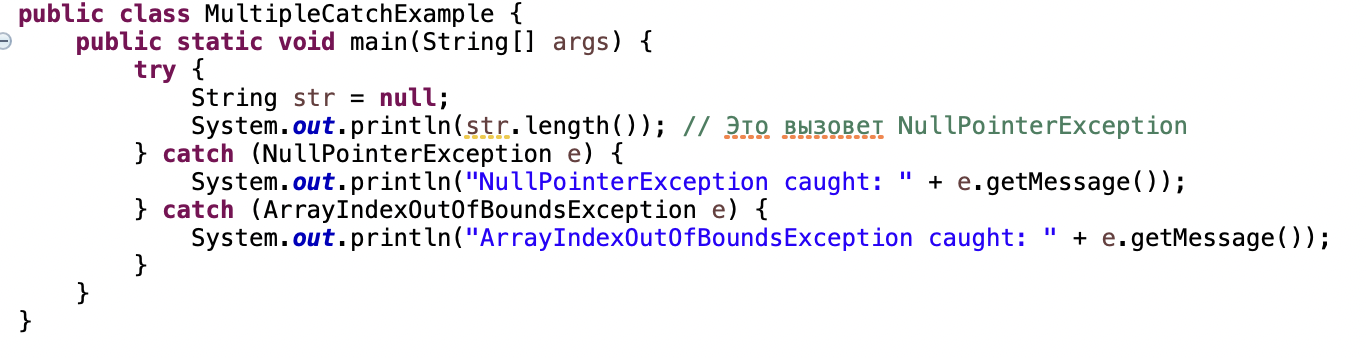


“В блоке try выполняется код, который может вызвать исключение. В блоке catch обрабатывается исключение, если оно возникло. Блок finally выполняется всегда, независимо от того, было ли исключение.”

4. Множественные блоки catch:

“Можно использовать несколько блоков catch для обработки различных типов исключений.”

Пример:

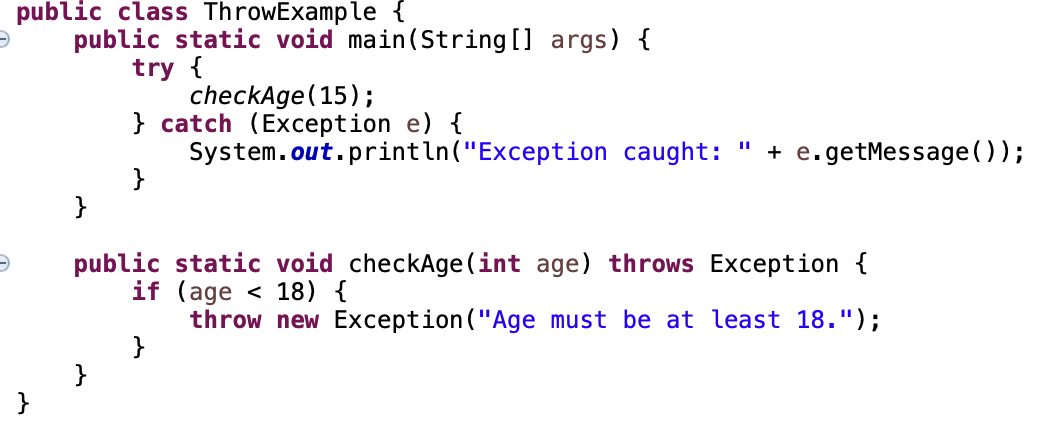


“В этом примере мы обрабатываем два типа исключений: NullPointerException и ArrayIndexOutOfBoundsException. Код внутри соответствующего блока catch будет выполнен, если возникнет соответствующее исключение.”

5. Вызов исключений с помощью throw:

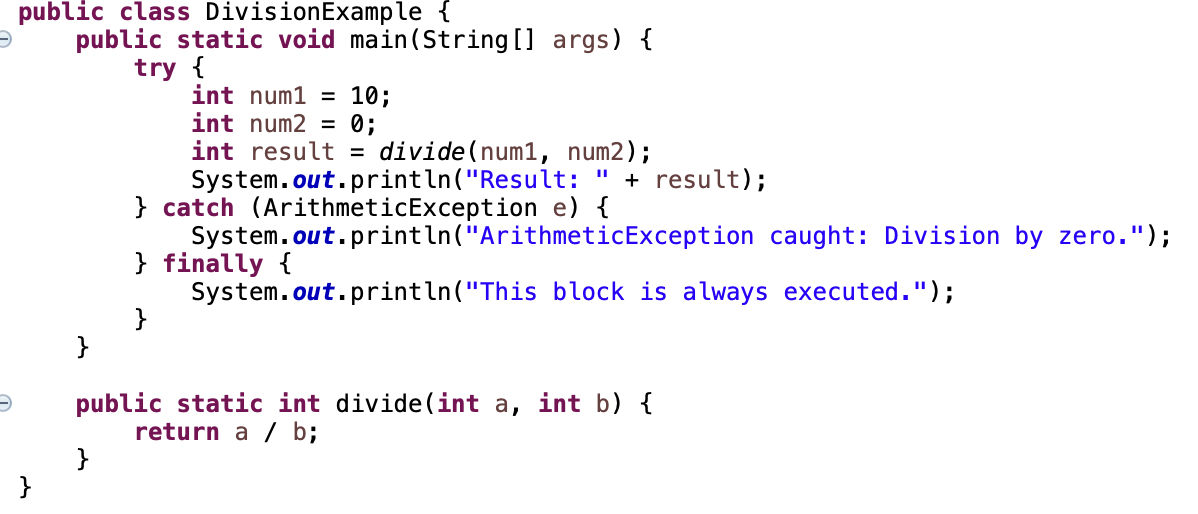
“Иногда бывает необходимо явно вызвать исключение с помощью ключевого слова throw.”

Пример:



“Метод checkAge вызывает исключение Exception, если переданный возраст меньше 18. Вызванное исключение обрабатывается в блоке catch в методе main.”

Пример программы:



“В этом примере метод divide вызывает ArithmeticException при попытке деления на ноль. Исключение обрабатывается в блоке catch в методе main, и блок finally выполняется всегда.”

**Некомпьютерная активность (10 минут):**

1. Обсуждение:

“Какие виды ошибок вы встречали в повседневной жизни?”

“Почему важно уметь обрабатывать ошибки?”

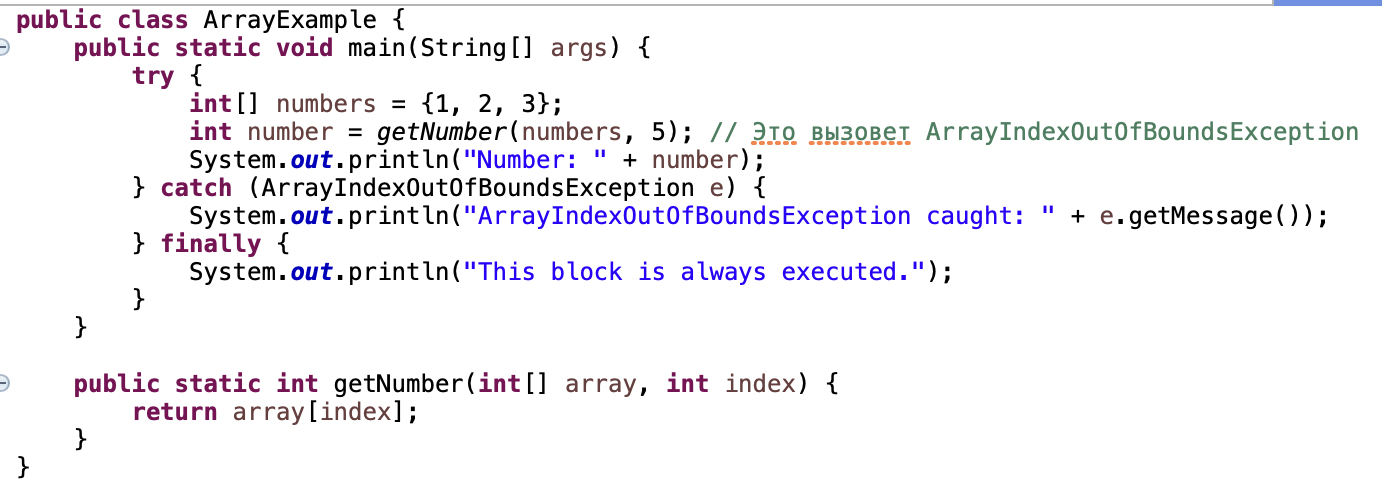
2. Групповая работа:

“Разделитесь на группы по 3-4 человека и обсудите, как можно использовать обработку исключений в ваших проектах.”

**Работа над проектом (25 минут):**

“Создайте новый проект и напишите программу, которая использует обработку исключений для корректной работы с массивами.”

Пример кода:

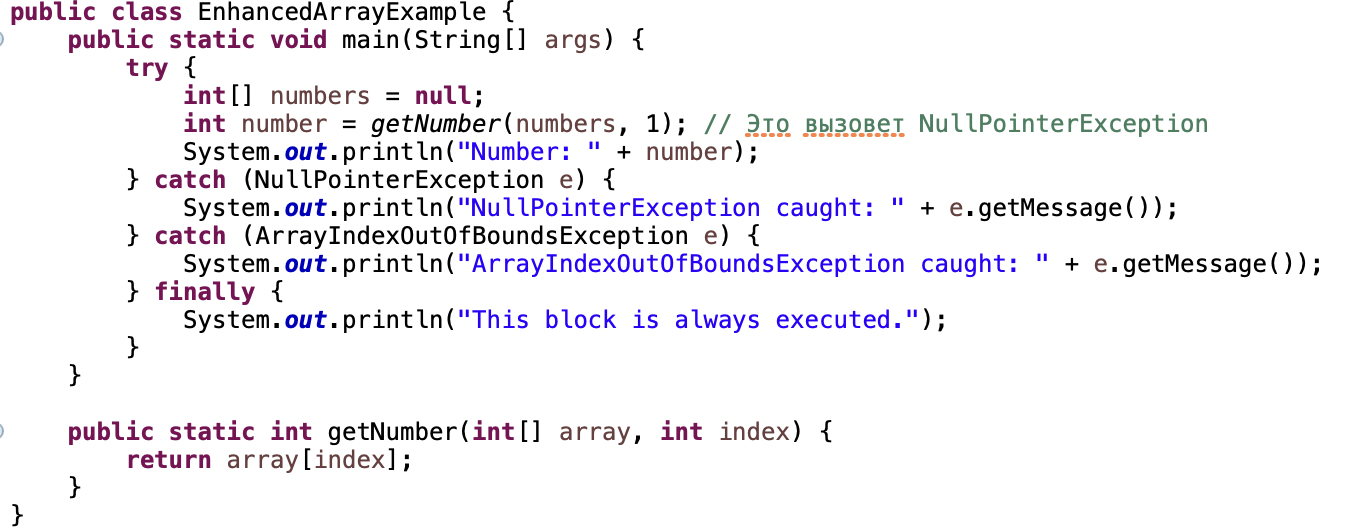


“Мы создали класс ArrayExample, который обрабатывает исключение ArrayIndexOutOfBoundsException при попытке доступа к несуществующему элементу массива.”

**Дополнительное задание:**

“Добавьте обработку возможных исключений, таких как NullPointerException и ArithmeticException.”

Пример кода:

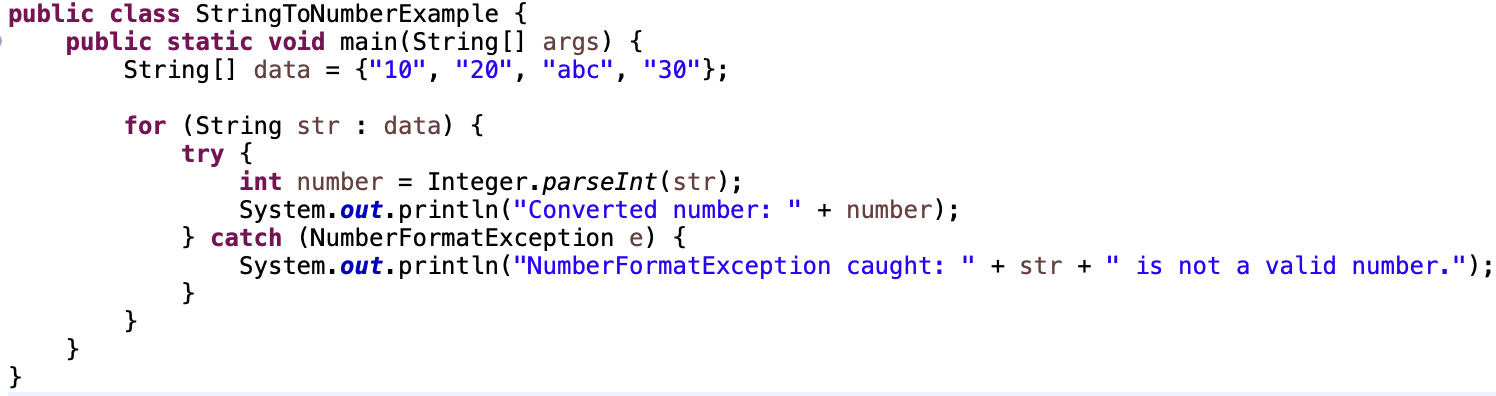


“Мы добавили обработку NullPointerException при попытке доступа к элементу массива через null ссылку и ArrayIndexOutOfBoundsException при выходе за пределы массива.”

**Проблемная задача (10 минут):**

“Напишите программу, которая считывает данные из массива строк, преобразует их в числа и обрабатывает возможные исключения.”

Пример кода:



“Мы создали класс StringToNumberExample, который обрабатывает NumberFormatException при попытке преобразования строки в число, если строка не является числом.”

Решение:

“Используйте метод Integer.parseInt() для преобразования строки в число и обрабатывайте возможное исключение NumberFormatException для корректной работы программы.”

**Рефлексия (10 минут):**

“Что нового вы узнали сегодня?”

“Почему важно уметь обрабатывать исключения в программах?”

“Какие трудности у вас возникли при выполнении заданий и как вы их преодолели?”

Обсуждение выполненного задания:

“Какие исключения вам удалось обработать в ваших программах? Как это улучшило их работу?”

Подведение итогов:

“Сегодня мы изучили исключения в Java, и научились их обрабатывать. Молодцы, вы отлично справились!”